Open Peer-to-Peer Systems over Blockchain and IPFS: an Agent Oriented Framework

Antonio Tenorio-Fornés
GRASIA, Universidad Complutense de Madrid
antoniotenorio@ucm.es

Samer Hassan
GRASIA, Universidad Complutense de Madrid
Berkman Klein Center at Harvard University
shassan@cyber.harvard.edu

Juan Pavón
GRASIA, Universidad Complutense de Madrid
jpavon@fdi.ucm.es

ABSTRACT

In recent years, the increasing concerns around the centralized cloud web services (e.g. privacy, governance, surveillance, security) have triggered the emergence of new distributed technologies, such as IPFS or the Blockchain. These innovations have tackled technical challenges that were unresolved until their appearance. Existing models of peer-to-peer systems need a revision to cover the spectrum of potential systems that can be now implemented as peer-to-peer systems. This work presents a framework to build these systems. It uses an agent-oriented approach in an open environment where agents have only partial information of the system data. The proposal covers data access, data discovery and data trust in peer-to-peer systems where different actors may interact. Moreover, the framework proposes a distributed architecture for these open systems, and provides guidelines to decide in which cases Blockchain technology may be required, or when other technologies may be sufficient.

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent systems**; • **Computer systems organization** → **Peer-to-peer architectures**;

KEYWORDS

Decentralization, Distributed Systems, P2P Systems, Framework, IPFS, Blockchain, Multi-Agent Systems

ACM Reference Format:
Antonio Tenorio-Fornés, Samer Hassan, and Juan Pavón. 2018. Open Peer-to-Peer Systems over Blockchain and IPFS: an Agent Oriented Framework. In *CryBlock'18: 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems , June 15, 2018, Munich, Germany.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3211933.3211937

1 INTRODUCTION

Nowadays, centralized cloud web services represent a large portion of the Internet [14, 24]. In the last years, there are increasing concerns on the multiple issues this situation arises, with respect to e.g. privacy [43], governance [20], legislation [14] , surveillance [36] or security [29].

Decentralized systems have tried to tackle these issues through interoperability [10, 44, 46] and federation [1, 10]. However, they are still hindered by several drawbacks, such as the existence of points of failure [41] and control [34], or the lack of interoperability of the data beyond specific applications [44].

Full decentralization would be certainly useful, especially for certain applications [30]. However, it was not until recently that some unresolved technical challenges [33, 45] have become more evident, which have been the driving forces to innovations such as Blockchain [39] and IPFS [3].

These new decentralized technologies enable multiple applications [4, 17, 18]. Nevertheless, there is a need for models and frameworks that explore how this technologies may be combined and what are their limitations and synergies in order to unveil the decentralization possibilities of recent innovations.

This work proposes a framework for the design and development of open distributed systems. The proposed model uses an agent-oriented approach, and, aiming to focus on real systems, the model assumes open systems (an open environment in which agents can join or leave freely [15, 23]) and where agents have partial information of the system data [22].

The rest of the paper is structured as follows. In Section 2, it defines the requirements of the considered systems, then it introduces the used decentralization technologies (Section 3). Section 4 discusses the consistency and search challenges of open distributed systems and provides design guidelines to asses whether those challenges may require using blockchain technology. Afterwards we proceed to provide an architecture to implement the proposed framework, in Section 5, where we use a distributed Questions and Answer system as example. The conclusions follows in Section 6.

2 SYSTEM REQUIREMENTS

This paper proposes a framework for distributed open systems with the following requirements:

(1) Open system: An open system is a system that enables external autonomous agents to freely join, leave and interact within it [15, 23]. Systems such as the World Wide Web (the Web) or Operating Systems are examples of open systems where new web servers or new programs can freely join and interact [5]. Such systems operate with certain degrees of uncertainty [7], as external actors can interfere in any given moment, and existing actors may leave. These open systems rely on interfaces, protocols and data types to enable the

CryBlock'18, June 15, 2018, Munich, Germany

ACM ISBN 978-1-4503-5838-5/18/06.
https://doi.org/10.1145/3211933.3211937

interactions within the system. The framework considers open systems to support the construction of heterogeneous and complex systems.

(2) Peer-to-peer system: Distributed systems are composed by a network of interconnected nodes that communicate and coordinate their actions (where such nodes may be e.g. computers or software agents) [12]. Systems such as the Web and P2P File sharing programs are distributed systems composed by web servers, and computers sharing files, respectively [5, 42]. While centralized systems depend on a single component for their operation, distributed systems are resilient to the disconnection of some of their components, e.g. if a web server is disconnected, the Web will still be a functional system. However, some distributed systems still depends on single components for parts of the system to work. For instance, if a web server disconnects, their web pages will become unavailable. This paper refers to *peer-to-peer systems* when referring to distributed systems that are independent from any single node.

(3) Agents with partial information: agents in open and distributed systems have access to just local knowledge of the system [22]. For instance, a web service may just have local knowledge about the resources it serves to the network. This model considers agents with local information that interact solely by 1) sharing new information in the system, 2) querying for information, and 3) responding to queries with their local information.

(4) Communication through a query protocol: Communication among agents of distributed systems is typically enabled through communication protocols [16, 19, 40]. These protocols enable agents to read (syntax) and understand (semantics) the messages involved in the communication. Moreover, they provide the sequence in which these messages must be exchanged. Although a wide variety of interactions can be enabled by communication protocols [40], this model proposes the use of a communication protocol that just allows to *share* information and to *query* for information (as other distributed systems do [47]).

The system proposes the use of queries that can be verified. Thus, an agent does not need to trust the agents providing the responses since these responses can be verified with regard to the query. This shared communication protocol also aims to enhance the interoperability of the proposed framework. The communication protocol is further described in Subsection 5.3.

3 DECENTRALIZATION TECHNOLOGY

This section introduces a technological background for the proposed framework. It describes Blockchain [39] and IPFS [3], the technological innovations that enable the development of new peer-to-peer systems previously unfeasible that this paper studies [3, 11, 25, 31, 35, 39] and some of its underlying concepts such as content-addressability and merkle linked structures.

Content Addressability In centralized and federated systems, content is frequently referred with addresses that include location information, the Uniform Resource Locators (URLs) [6]. However, references to content can also be independent from their location, using Universal Resource Identifiers (URIs) [26]. In peer-to-peer systems, agents cannot rely on the location of other agents for accessing content, because the content could be provided by any agent. The hash[1] of any content can be used as its URI. Thus, these hash URIs are used in multiple distributed systems such as IPFS to build scalable content-addressable networks [3, 27, 38, 42].

Merkle Links and Structures The use of hash values (see previous subsection) to reference data in data structures was first introduced by [37]. Complex data structures can use these links (See Figure 1 for a Merkle structure example). This Merkle linked structures are key to build technologies such as Git [35], Blockchain [39] and IPFS [3] among others. Section 5.2 propose the use of these structures for the data representation of the system.

Blockchain Blockchain was the first technology that enabled a fully distributed digital currency[39]. It uses a Merkle Linked list of blocks of transactions (a Blockchain) to build a distributed ledger of transactions. It made computationally difficult to propose a candidate for the next block in the distributed ledger and incentives nodes to try to build those candidates with valid transactions. Then, the protocol requires that honest nodes will consider the largest chain they have observed in a given time as the actual ledger to trust. Therefore, in order to forge a blockchain, an actor would need half of the computing power of the system. Section 4.3 proposes the use of Blockchain to provide consistency to open distributed systems.

IPFS Some peer-to-peer systems like P2P sharing software [42] use hash of the content to address it. Other technologies such as Git use complex Merkle-Linked Structures[35]. IPFS integrates both the use of complex Merkle-Linked structure with the data-addressability of P2P file sharing systems. The content is distributed over a peer-to-peer network. Section 5.1 proposes the use of IPFS for the storage and distribution of data in the framework.

4 CHALLENGES OF DISTRIBUTED OPEN SYSTEMS: CONSISTENCY AND SEARCH

Data discovery in decentralized open systems is a challenge [23]. This section frames this challenge in the following three subsections:

- CAP Theorem [8] (Subsection 4.1) introduces the compromises between *C*onsistency, *A*vailability and *P*artition resistance in distributed systems.
- CALM Principle [2] (Subsection 4.2) provides analysis tools to assess whether a distributed system (or search) needs coordination
- Blockchain technology provides the first peer-to-peer coordination mechanism for distributed systems requiring trustless strong consistency such as cryptocurrencies (Subsection 4.3).

[1] Hash functions are one-way collision-free functions, i.e. functions that, given their output, the probability to guess which input produced it is negligible.

4.1 CAP Theorem

CAP Theorem [8] states that a networked data system can only provide two out of these three desirable properties:

(1) *C*onsistency: The requests of the distributed system behaves as if handled by a single node with updated information.
(2) *A*vailability: every request should be responded.
(3) *P*artition resistance: the system is able to operate in presence of network partitions.

Given that the framework considers open systems, the *P*artition resistance is a needed property for our proposal. Therefore, one of the most important design decisions for the systems built within the framework is to find the best balance between *C*onsistency and *A*vailability.

4.2 CALM Principle

Discovering information within a distributed network is a challenge, since the information may be scattered among many nodes. In fact, some requests are impossible to resolve within distributed open systems. Intuitively, in an open system we cannot know all the data. Therefore, queries that need to take into account all the information of the system such as those counting the data that satisfy some constraints are impossible to resolve.

*C*onsistency *A*s *L*ogical *M*onotonicity (CALM) principle provides a tool to describe which queries can be resolved in a distributed system without coordination [2]. In a system with logical monotonicity, a true statement remains to be true with the addition of new axioms. The results of a distributed search will be consistent if the query is monotonic, i.e. if considering new information, the results cannot change.

The designer of a distributed system can check the monotonicity of its queries as follows:

(1) A sufficient condition for monotonicity is order independence [2]. For instance, the double spend problem where an agent tries to spend "the same coin" twice in distributed currencies arises from the impossibility to know which payment was done earlier without a coordination mechanism: it is a non-monotonic problem.
(2) If adding new information may change the validity of a response to a query, then it is non-monotonic, e.g. the search of the most voted answer in a Q&A system is non-monotonic, since new votes to an alternative answer would change the response.
(3) Formal analysis of the queries can be done to assess logical monotonicity [2].

Non-monotonic queries produce non consistent results in distributed systems without coordination (e.g. the double spending problem). Thus, in the presence of non-monotonic queries, the designer should decide on the consistency requirements of the system.

Guideline 1. *Monotonic queries can be implemented without using Blockchain or other coordination technologies.*

If inconsistent behaviour, like missing some votes in a Q&A system, is acceptable for the system, then coordination mechanisms are still not needed. If inconsistent behaviour is unacceptable, for instance the double-spend problem in distributed currencies then a coordination mechanism is needed. Blockchain technology is a coordination mechanism that provides consistency while maintaining the system distributed.

Guideline 2. *Consistency requirements are a design decision. If inconsistent behaviour is acceptable for the non-monotonic queries of the system, coordination technologies such as Blockchain are not required.*

Guideline 3. *The non-monotonic queries of the system with strong consistency requirements should be supported by a coordination technology such as Blockchain.*

4.3 Blockchain for distributed consistency

Blockchain was indeed proposed as a way of coordinating a non-monotonic problem for an open distributed system: the double-expend problem, where a malicious actor may try to simultaneously pay twice with the same coin in a distributed payment system. The order in which these payments are processed matters, since the second payment would not be considered valid.

Recording and validating the interactions of a distributed system in a Blockchain (a distributed ledger) provides consistency for non-monotonic systems. Note that in open systems, full partition recoveries and explicit partition management are not expected, and therefore solutions that rely on them such as CRDTs [8] are not applicable.

5 ARCHITECTURE

The architecture of the proposed framework is presented with an example of the implementation of a simple Questions and Answers (Q&A) system, similar to the popular Stack Exchange[2] and its most famous instance Stack Overflow[3].

The architecture uses IPFS as a distributed data store, public-key identities for data trust, and a generic P2P network for communication. Based in the design guidelines presented in previous section, it proposes the use of Blockchain technology when strong consistency is a requirement. The discussion of data access, data trust and data discovery of the system structures the presentation of this open and distributed architecture.

5.1 Tackling Data access

Traditional Q&A systems such as Stack Exchange use a location-centric model for data access. In these systems, specific nodes called hosts are responsible for data provision and are trusted for providing the requested data. For instance, when a user has a programming question, she may search in Stack Overflow website for answers.

Our architecture proposes the use of content-addressable data as alternative to distribute the systems data provision and access. Concretely, it proposes the use of Merkle-linked structures distributed over the IPFS newtwork. Structuring the information as IPFS objects provide both the Merkle-linked structure and the data-addressability of the information [3]. The nodes of this structures are objects composed 1) by key-value pairs representing their attributes and 2) by named directed Merkle-links to other nodes. A

[2] https://stackexchange.com
[3] https://stackoverflow.com

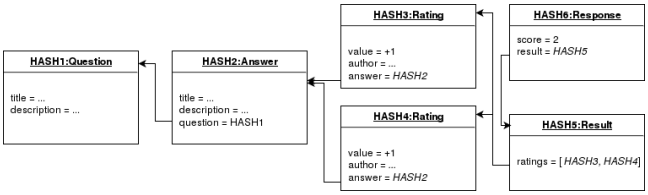

Figure 1: Merkle linked data of an example Question and Answers system (such as Stack Overflow)

representation of linked questions, answers, and votes of a Q&A system is depicted in Figure 1.

The data will be distributed through IPFS. Any agent with system information can act as *data provider* of that information. Moreover, specialized provider agents can be deployed to ensure the availability of information.

5.2 Tackling Data Trust

Both centralized and federated systems use direct communication with trusted hosts to obtain trustworthy data. For instance, centralized Q&A systems trust a web server. However, peer-to-peer alternatives can be explored to enable other nodes to provide trusted data.

This architecture proposes trusting cryptographic identities instead of hosts for providing trustworthy data. Data signed by valid identities is then trusted in the system. In order to enable an easier integration with other parts of the framework, the architecture suggest the use of IPNS [3] or Ethereum [9] identity infrastructure.

Considering our running example, questions, answers and votes would be signed by their authors. Following Stack Exchange rules, new identities can ask questions or provide answers. Thus, in a distributed implementation, any identity could sign questions and answers. However, Stack Exchange requires at least 15 reputation points to be able to vote. Thus, our system would only trust a vote signed by an identity with at least that reputation. Reputation is given for the quality of the user's contributions, for instance, each positive vote in a question or answer gives the user 5 reputation points (as in Stack Exchange).

Thus, the information needed to trust an answer with one vote would be: 1) the question, signed by any identity, 2) the vote signed by an identity that have signed questions and answers that have received three valid votes. 3) recursively validate the new three votes.

With this example we observe that although it is possible to replicate the logic of some centralized systems, the complexity and size of the data needed to trust some information may not be trivial.

Non-monotonic searches (see Section 4.2), such as getting exact number of votes of a question or knowing if a question was reported as spam, may need the use of a blockchain as coordination mechanism. For instance, votes may be registered in a blockchain, enabling verifiable responses to non-monotonic searches. This work proposes the use of Ethereum [9] for the development of blockchain-based smart contracts that govern the logic and consistency of such systems.

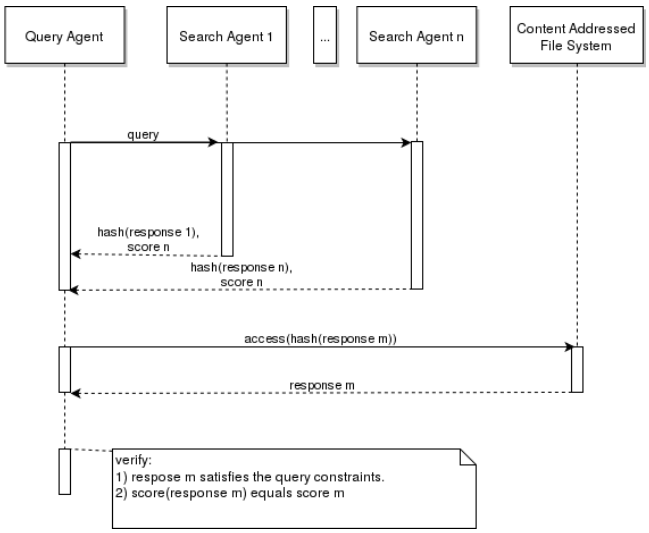

Figure 2: Distributed Discovery Protocol UML Sequence Diagram

5.3 A Trustless Distributed Data Discovery Protocol

The protocol proposes the definitions of queries as constraints to be satisfied by data responses. For instance, a question in a Q&A system can be searched and constraints over its content (e.g. it contains a list of words) and over its structure (e.g. has at least one answer) can be requested.

In addition, the protocol allows the definition of *score* functions for the responses satisfying the queries constraints. This is later used to rank the responses. For instance, the number of votes can be used to sort the searches.

Finally, the protocol interactions (Figure 2) are defined as follows:

(1) An agent sends a query consisting of the constraints and score function.
(2) Any agent can reply with a content-centric link to the data satisfying the query and the result of the score function applied to the data.
(3) The agent can then access the data of the responses. The response can be verified to satisfy the constraints and to score the provided score value.

The protocol as described above has the following advantages:

(1) Lightweight communication: responses consist of a short link and a numeric value. Their length is then a few bytes long while they may represent complex large data structures.
(2) Early distributed comparison/verification: Allows the comparison of responses before even knowing the responses content, in a trustless manner.
(3) Trustless ranking and validity: Responses can be checked to satisfy both the constraints (and thus their validity) and the score function (and thus their ranking with respect to other responses).

The proposed implementation of the protocol relies in: 1) IPFS merkle-linked objects to represent the data and provide the responses. 2) Javascript pure functions to express query constraints and score functions, using the JavaScript implementation of IPFS, and 3) A bus model for distributed systems communication [28] over IPFS pub-sub channels.

6 DISCUSSION AND CONCLUSIONS

This work presents a framework to build peer-to-peer open systems as a multi-agent systems. It enables the data access, data discovery and data trust in a decentralized infrastructure, targeting some of the challenges of fully distributed systems.

The framework studies recent technologies such as IPFS and Blockchain that enable previously unfeasible distributed systems (such as crypto-currencies[39]). It proposes design guidelines to asses whether a coordination tool is needed to provide strong consistency in distributed open system and proposes the use of Blockchain for such cases.

A distributed architecture is proposed for the implementation of the studied systems. IPFS and its merkle linked structures are proposed for data representation and distribution, Public key cryptography is used to provide trust to the distributed data, and Ethereum Blockchain technology is proposed as coordination tool to support the non-monotonic consistency requirements of the systems. A simple channelled flooding algorithm over the IPFS infrastructure is proposed as sample communication infrastructure. The framework also proposes the use of a query communication protocol which enables data discovery in open distributed systems and support both ranked responses and trust-less verification of the responses.

Thus, the presented framework supports the design and implementation of peer-to-peer systems using the innovations introduced by Blockchain and IPFS. The theoretical limitations of these technologies inform the propposed design guidelines, providing tools to asses whether using Blockchain is recommended for the system.

The proposal inherits the challenges and limitations of Blockchain-based and distributed technology such as privacy [13, 21] and sustainability [11]. Moreover, some security issues such as *sybil attacks* [39] and *generation attacks* [32] deserves special consideration in the systems designed with the framework. Still, distributed technologies most frequently provide better privacy than their centralized counterparts [46].

The performance and efficiency of the proposed framework remains to be studied in future work. The deployment of specialized agents, such as search agents for specific applications, or the proposal of improved network topologies and protocols are some of the performance improvement opportunities to explore.

The implementation of new open decentralized systems as interoperable multi-agent systems may enable the growth of a new family of complex and heterogeneous peer-to-peer systems. This paper have introduced a framework to build these systems using the potentials of new decentralizing technologies.

7 ACKNOWLEDGMENTS

This work was partially supported by the project P2P Models (https://p2pmodels.eu) funded by the European Research Council ERC-2017-STG (grant no.: 759207), and ColoSAAL (http://grasia.fdi.ucm.es/colosaal/), funded by the Spanish Ministry of Economy and Competitiveness (TIN2014-57028-R).

REFERENCES

[1] Mansour Alsaleh and Carlisle Adams. 2006. Enhancing consumer privacy in the liberty alliance identity federation and web services frameworks. In *International Workshop on Privacy Enhancing Technologies*. Springer, Berlin, Heidelberg, 59–77.
[2] Peter Alvaro, Neil Conway, Joseph M Hellerstein, and William R Marczak. 2011. Consistency Analysis in Bloom: a CALM and Collected Approach.. In *CIDR 2011 - 5th Biennial Conference on Innovative Data Systems Research, Conference Proceedings*. Asilomar, California, 249–260.
[3] Juan Benet. 2014. IPFS-content addressed, versioned, P2P file system. *arXiv preprint arXiv:1407.3561* (2014).
[4] Brendan Benshoof, Andrew Rosen, Anu G Bourgeois, and Robert W Harrison. 2016. Distributed Decentralized Domain Name Service. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*. IEEE, Chicago, IL, USA, 1279–1287.
[5] Tim Berners-Lee. 2010. Long live the web. *Scientific American* 303, 6 (2010), 80–85.
[6] Tim Berners-Lee, Larry Masinter, and Mark McCahill. 1994. *Uniform resource locators (URL)*. Technical Report.
[7] Shahriar Bijani and David Robertson. 2014. A review of attacks and security approaches in open multi-agent systems. *Artificial Intelligence Review* (2014), 1–30.
[8] Eric Brewer. 2012. CAP twelve years later: How the" rules" have changed. *Computer* 45, 2 (2012), 23–29.
[9] Vitalik Buterin. 2014. Ethereum: A next-generation smart contract and decentralized application platform. *URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper* (2014).
[10] Florencio Cabello, Marta G Franco, and Alex Haché. 2013. The Social Web beyond'Walled Gardens': Interoperability, Federation and the Case of Lorea/n-1. *PsychNology Journal* 11, 1 (2013), 43–65.
[11] Bram Cohen. 2003. Incentives build robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer systems*, Vol. 6. 68–72.
[12] George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. 2011. *Distributed Systems: Concepts and Design* (5th ed.). Addison-Wesley Publishing Company, USA.
[13] Primavera De Filippi. 2016. The interplay between decentralization and privacy: the case of blockchain technologies. (2016).
[14] Primavera De Filippi and Smari McCarthy. 2012. Cloud Computing: Centralization and Data Sovereignty. *European Journal of Law and Technology* 3, 2 (2012). https://ssrn.com/abstract=2167372
[15] Yves Demazeau and AC Rocha Costa. 1996. Populations and publishers in open multi-agent systems. In *Proceedings of the 1st National Symposium on Parallel and Distributed AI*. 1–13.
[16] Ksenia Ermoshina, Francesca Musiani, and Harry Halpin. 2016. End-to-end encrypted messaging protocols: An overview. In *International Conference on Internet Science*. Springer, 244–254.
[17] José G. Faísca and José Q. Rogado. 2016. Decentralized Semantic Identity. In *Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS 2016)*. ACM, New York, NY, USA, 177–180. https://doi.org/10.1145/2993318.2993348
[18] J. G. Faisca and J. Q. Rogado. 2016. Personal cloud interoperability. In *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. 1–3. https://doi.org/10.1109/WoWMoM.2016.7523546
[19] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. 1994. KQML as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*. ACM, 456–463.
[20] Mayo Fuster Morell. 2010. *Governance of online creation communities: Provision of infrastructure for the building of digital commons*. Ph.D. Dissertation.
[21] B. Greschbach, G. Kreitz, and S. Buchegger. 2012. The devil is in the metadata—New privacy challenges in Decentralised Online Social Networks. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. 333–339. https://doi.org/10.1109/PerComW.2012.6197506
[22] Frans CA Groen, Matthijs TJ Spaan, Jelle R Kok, and Gregor Pavlin. 2005. Real world multi-agent systems: information sharing, coordination and planning. In *International Tbilisi Symposium on Logic, Language, and Computation*. Springer, 154–165.
[23] Carl Hewitt and Peter De Jong. 1984. Open systems. In *On Conceptual Modelling*. Springer, 147–164.
[24] Benjamin Mako Hill. 2008. Franklin Street statement on freedom and network services. *Autonomo. us* (2008). https://web-beta.archive.org/web/20151006113744/http://autonomo.us/2008/07/14/franklin-street-statement/
[25] Luis-Daniel Ibáñez, Elena Simperl, Fabien Gandon, and Henry Story. 2017. Redecentralizing the web with distributed ledgers. *IEEE Intelligent Systems* 32, 1 (2017), 92–95.

[26] Ian Jacobs. 2004. URIs, Addressability, and the use of HTTP GET and POST. *World Wide Web Consortium, TAG Finding* 13 (2004), 28.
[27] Jan Janak, Jae Woo Lee, and Henning Schulzrinne. 2011. GRAND: Git Revisions As Named Data. (2011).
[28] Kyungkoo Jun, Ladislau Boloni, Krzysztof Palacz, and Dan C Marinescu. 2000. Agent-based resource discovery. In *Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th*. IEEE, 43–52.
[29] Balachandra Reddy Kandukuri, Atanu Rakshit, et al. 2009. Cloud security issues. In *Services Computing, 2009. SCC'09. IEEE International Conference on*. IEEE, 517–520.
[30] Anne-Marie Kermarrec. 2013. Towards a personalized Internet: a case for a full decentralization. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 371, 1987 (2013), 20120380.
[31] L Kissner and Ben Laurie. 2009. General verifiable federation. *Google white paper* (2009).
[32] Protocol Labs. 2017. Filecoin: A Decentralized Storage Network. https://filecoin.io/filecoin.pdf
[33] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4, 3 (1982), 382–401.
[34] Ladar Levison. 2014. Secrets, lies and Snowden's email: why I was forced to shut down Lavabit. *The Guardian* (2014).
[35] Jon Loeliger and Matthew McCullough. 2012. *Version Control with Git: Powerful tools and techniques for collaborative software development.* " O'Reilly Media, Inc.".
[36] David Lyon. 2014. Surveillance, Snowden, and big data: Capacities, consequences, critique. *Big Data & Society* 1, 2 (2014), 2053951714541861.
[37] Ralph C Merkle. 1987. A digital signature based on a conventional encryption function. In *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 369–378.
[38] Robert Morris, M Frans Kaashoek, David Karger, Hari Balakrishnan, Ion Stoica, David Liben-Nowell, and Frank Dabek. 2003. Chord: A scalable peer-to-peer look-up protocol for internet applications. *IEEE/ACM Transactions On Networking* 11, 1 (2003), 17–32.
[39] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).
[40] Stefan Poslad. 2007. Specifying protocols for multi-agent systems interaction. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 2, 4 (2007), 15.
[41] Enayat Rajabi, Salvador Sanchez-Alonso, and Miguel-Angel Sicilia. 2014. Analyzing broken links on the web of data: an experiment with DBpedia. *Journal of the Association for Information Science and Technology* 65, 8 (2014), 1721–1727.
[42] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. 2001. *A scalable content-addressable network*. Vol. 31. ACM.
[43] Ira Rubinstein and Joris Van Hoboken. 2014. Privacy and security in the cloud: some realism about technical solutions to transnational surveillance in the post-Snowden era. (2014).
[44] Andrei Vlad Sambra, Essam Mansour, Sandro Hawke, Maged Zereba, Nicola Greco, Abdurrahman Ghanem, Dmitri Zagidulin, Ashraf Aboulnaga, and Tim Berners-Lee. [n. d.]. Solid: A Platform for Decentralized Social Applications Based on Linked Data. ([n. d.]). Technical report.
[45] Matthias Wachs, Martin Schanzenbach, and Christian Grothoff. 2014. On the feasibility of a censorship resistant decentralized name system. In *Foundations and Practice of Security*. Springer, 19–30.
[46] Ching-man Au Yeung, Ilaria Liccardi, Kanghao Lu, Oshani Seneviratne, and Tim Berners-Lee. 2009. Decentralization: The future of online social networking. In *W3C Workshop on the Future of Social Networking Position Papers*, Vol. 2. 2–7.
[47] H. Zarzour and M. Sellami. 2013. p2pCoSU: A P2P Sparql/update for collaborative authoring of triple-stores. In *2013 11th International Symposium on Programming and Systems (ISPS)*. 128–136. https://doi.org/10.1109/ISPS.2013.6581478